Eligible Earliest Deadline First: Server-Based Scheduling for Master-Slave Industrial Wireless Networks

Michael Short
School of Science, Engineering and Design, Teesside University, Middlesbrough TS1 3BA, UK.

m.short@tees.ac.uk

Abstract: Industrial automation and control systems are increasingly deployed using wireless networks in master-slave, star-type configurations that employ a slotted timeline schedule. In this paper, the scheduling of (re)transmissions to meet real-time constraints in the presence of non-uniform interference in such networks is considered. As packet losses often occur in correlated bursts, it is often useful to insert gaps before attempting retransmissions. In this paper, a quantum Earliest Deadline First (EDF) scheduling framework entitled 'Eligible EDF' is suggested for assigning (re)transmissions to available timeline slots by the master node. A simple but effective server strategy is introduced to reclaim unused channel utilization and replenish failed slave transmissions, a strategy which prevents cascading failures and naturally introduces retransmission gaps. Analysis and examples illustrate the effectiveness of the proposed method. Specifically, the proposed framework gives a timely throughput of 99.81% of the timely throughput that is optimally achievable using a clairvoyant scheduler.

Keywords—Master-Slave Communication Systems; Industrial Wireless Networks; Real-Time Scheduling; Error-Recovery.

1.0 Introduction

The use of wireless communication systems in monitoring and control applications such as factory automation, smart grid and process control has been increasing at a steady rate in recent years [1-10]. This is partly due to a number of recent improvements in the technology, reliability and cost of wireless communication equipment; see for example the survey papers by Akyildiz et al. [9], Rashid and Rehmani [10], Christin et al. [11] and Ajith Kumar et al. [12]. Research on routing protocols to minimize energy consumption [13], low-overhead operating systems for sensor/actuator nodes [14] and novel energy-scavenging techniques [15] have all made contributions that help increase the battery life of devices and hence maximize network up-time.

Wireless systems have the distinct advantage of reducing equipment installation complexity through the lack of a need for wiring and harnessing, enabling easier trouble-shooting and system re-configuration; this reduces the long-term maintenance requirements associated with wired systems [1–4]. However, the use of wireless technologies in these applications is not without problems, which include out-of-order packet transmissions, high levels of packet jitter and high probabilities of packet losses. These problems are especially problematic in control applications that can have strict timing constraints [1-4]. Some of these problems can be ameliorated to a certain extent by careful planning of node locations [5], using redundant ratio transceivers [8] or by using higher level (application layer) compensation techniques [16]. The use of master-slave (request-response) type architectures and their close Time Division Multiple

Access (TDMA)-based variants connected in star or mesh topologies is also a popular method to help reduce such problems [4-6][17-18]. The focus of this paper is mainly upon industrial master-slave networks connected in star topologies, in which messages are scheduled using a master-controlled slotted timeline approach or a master-controlled TDMA 'superframe' approach. Typically, devices in such networks will operate with equipment using frequencies in the unlicensed Instrumentation, Scientific and Medial (ISM) radio-band and although error detection and correction techniques are routinely employed, they can suffer from large amounts of interference in an industrial environment [4][5][7].

Since interference can affect the successful delivery of packets, care must be taken to ensure that the control or monitoring devices do not operate with an inconsistent view of the physical process. Aside from using redundancy in the frequency spectrum (e.g. [7]), a key method to increase the reliability of packet delivery (and hence transaction reliability) is to allow some spare or 'slack' time in the communications schedule and employ retransmissions in case of detected errors (temporal redundancy). In this paper, a scenario in which master-slave scheduling of request-response transactions takes place in the presence of non-uniform and bursty interference is considered, and in situations in which more complex patterns of real-time communication than those afforded by a simple cyclic schedule are required to take place. Specifically, the case in which transactions are under the control of a master node using the Earliest Deadline First (EDF) real-time scheduling algorithm is considered, for transactions having relative deadlines equal to their periods. In such circumstances if the scheduler has accurate knowledge of the upcoming channel error state *prior* to scheduling a transaction, then scheduling the *feasible* transaction with earliest deadline has optimal properties [19]. However, this implies the scheduler must be clairvoyant and unfortunately, such a 'Feasible EDF' scheme cannot be implemented in practice [19]. The use of on-line (statistical) estimates of the channel state or probe packets is required prior to making scheduling decisions; neither technique can be 100% accurate due to the nature of the problem, and both consume unwanted additional node and/or bandwidth overheads [19].

A simple but practical alternative to Feasible EDF is explored in this paper. The proposed technique – entitled 'Eligible EDF' - does not attempt to estimate or predict the channel state in either the master or the slave nodes, and it does not use probe packets. Transactions are scheduled using EDF without explicit knowledge of the channel state and hence errors will occur; re-scheduling of failed transactions in the case of packet losses is handled by a simple but effective replenishment strategy controlled by a server. This server reclaims unused channel utilization to replenish failed slave transactions, a strategy which prevents 'domino' deadline failures affecting other pending transactions. Under some mild assumptions related to the boundedness of the packet size and the worst-case slave turnaround times, this situation is well represented as a Quantum scheduling instance for which a very simple optimal server is known. By replenishing failed transactions within this server according to their absolute deadlines, retransmissions are handled effectively and a close approximation of Feasible EDF is obtained. This occurs since retransmission attempts are temporarily separated by a minimum gap as a natural by-product of the server operation, and infeasible channels are effectively polled (within the main schedule) until they become feasible. This helps to de-correlate packet losses and increases the reliability of packet delivery without starvation of feasible channels occurring. Together, this seems to give a flexible and robust means to schedule transmissions in a master-slave wireless network. Computational experimental results indicate that the achieved timely throughput is very close ($\approx$ 99.81%) to the optimal timely throughput which can be obtained using a clairvoyant Feasible EDF scheduler having perfect channel state estimation.

The remainder of this paper is organized as follows. Section 2 of the paper describes related work on industrial wireless communications with specific focus upon scheduling in master-slave configurations. Section 3 describes the assumed models of the communication system and channel errors. Section 4 describes the proposed scheduling technique. Section 5 presents a series of computational experiments using simulations to evaluate the technique, and presents analysis of the results obtained. Conclusions and areas for further work are considered in Section 6.

2.0 Related Work

As mentioned in the introduction, this paper assumes a master-slave configuration for the wireless network, which has a (possibly hierarchical) star-type or mesh-type topology. In the master-slave star connected approach a single (typically central) master node has control over the overall communication by sending out requests to slaves to elicit responses in a static (pre-determined) or dynamic cyclic order [4][17][18][20]. A number of technologies for wireless factory automation applications have been surveyed and compared with respect to attributes such as flexibility, security and Quality of Service (QoS) [11][12]. The technologies surveyed include the Wireless Interface for Sensor and Actuators (WISA), WirelessHART, ISA100.11a, Zig-Bee, ZigBee PRO, and 802.15.4e Factory Automation MAC Layer. Although each technology can typically configured in one of several different ways, a popular configuration for Supervisory Control And Data Acquisition (SCADA), Distributed Control Systems (DCSs), process monitoring and factory automation applications is a (possibly hierarchical) star configuration, such as is depicted in Figure 1.

The figure shows several workcells, each containing a master node and multiple slave nodes. Typically, the master node in each workcell (which would consist of Programmable Logic Controllers (PLCs), Programmable Automation Controllers (PACs) or other embedded automation or control devices) would communicate with slave communication nodes (which would consist of remote I/O and sub-control units) via short/medium-range packet radio systems. Normally the designation of master node in each workcell is relatively easy to assign, due to the differentiation between the automation/control device and the sub-control or I/O units. In some cases, the master node may even be an access point, with the main automation and control algorithms hosted on the network manager; such an approach is utilized in the Emerson Delta-V(c) commercial DCS, which uses a proprietary Wireless Interface for Sensors and Actuations (WISA) access points as master nodes. The slave nodes can be considered fixed in most situations, but may be mobile in some cases (such as interfaces to autonomous vehicles).

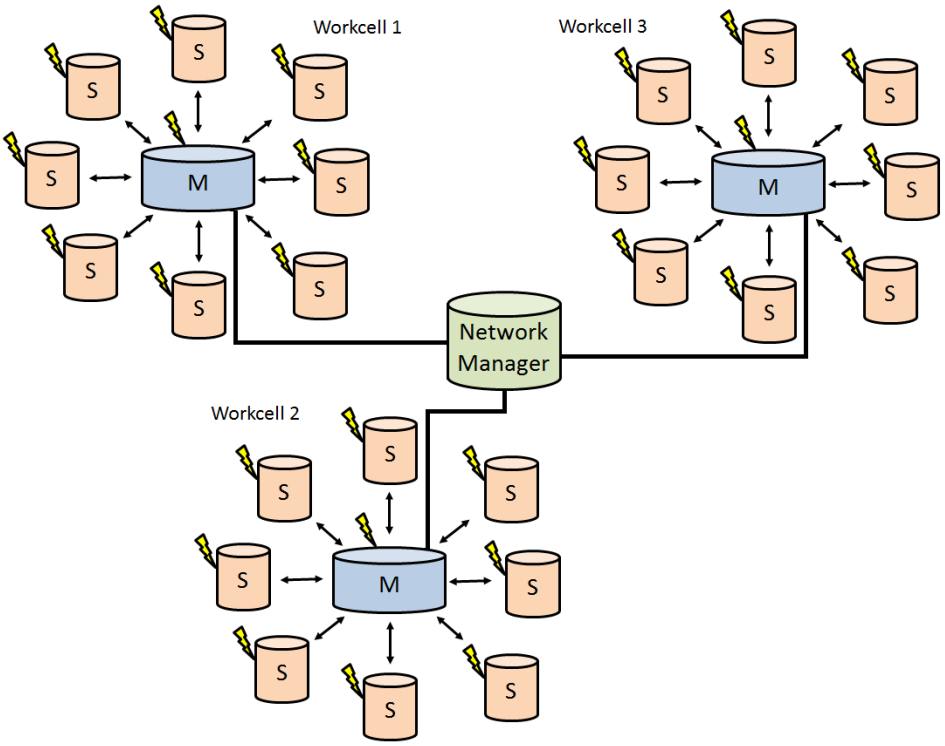

Fig. 1: Typical hierachical star-type Master-Slave architecture employed in factory automation

The architecture above is a good fit for many industrial applications using application level protocols such as Modbus, which operates on a Client-Server (request-response transaction) basis [21]. Packet radio systems are commonly used to upgrade legacy automation or SCADA devices and commercial products such as wireless TIA-485 modems are now commonplace in industrial applications [6]. In addition, low-cost small form factor interfaces such as XBee© range of modules provide transparent point-to-point or point-to-multipoint (star) connectivity for devices with simple Universal Asynchronous Receiver Transmitter (UART) interfaces. The modules support serial data rates of ≈ 250 kbps and enable microcontroller families such as those used by the Arduino© range of open source hardware modules to become part of a wireless sensor/actuator network used in automation applications with ease. The master nodes in each workcell are connected (again, typically using star-type topology) to a central network manager or control station, typically using a wired communication backbone. Note that the topology of Figure 1 is not just restricted to SCADA-type applications and also applicable to several other application domains: for example, a cellular data network where each access point (master node) is connected via a backhaul network to a service provider and the slaves are user handsets [19]. The choice of master node designation is again enforced by the nature of the application infrastructure, since the access points consist of especially build antenna towers.

The use of master-driven request-response ('polling') type architectures within a star topology industrial wireless network is very popular [4-6][17-18]. The focus of this paper is mainly upon industrial master-slave networks using such a request-response approach for messages having non-uniform period and deadline requirements, along with support for occasional non-periodic traffic. The proposed method is also applicable to more generic uplink/downlink scheduling in which messages are dynamically scheduled into superframe timeline slots by a centralised access point. A

superframe approach is used to orchestrate the timing of communicating devices on the network [20]. The superframe typically starts with the transmission of a beacon by the master node (access point) followed by a finite number of transmission slots (typically of duration 5 ms - 15 ms each for industrial protocols [11]). In between the beacon and the commencement of the first transmission slot, some control information to allocate slots to slave devices is normally present.

Typically, the wireless networks employed in figure 1 will utilize devices operating with frequencies in the unlicensed ISM-band. Although error detection and correction techniques are routinely employed, the networks can suffer from large amounts of non-uniform and bursty interference [4][5][7]. In case of failed transmissions, various retransmission strategies can be considered, some of which have been shown to be more successful than others. When simple cyclic polling of slaves within a superframe is employed, as found by Gamba et al. the insertion of a gap in the schedule before attempting re-transmission across a failed channel can be beneficial as it allows some time for the burst to subside [4]. Since interference in a wireless network is not necessarily uniform across all nodes simultaneously, the most successful strategies for simple cyclic polling have been found to be variants of those which temporarily defer a retransmission attempt to a particular slave node in favour of attempting to progress the cyclic schedule by communicating with other slave nodes in the intervening time [4].

A cross-layer source-aware adaptation of TDMA scheduling for star and mesh networks is presented by Shen et al. [22]. In their work, a time-slotted TDMA access mechanism where both uplink and/or downlink transmissions are scheduled in a superframe by a base (master) station is considered. Interference is assumed to be both external and internal to the network; in addition to channel fading, it is assumed that nodes can transmit on one of multiple channels (typically 16), and any nodes sharing a radio channel may cause interference if they are located in close enough proximity with one another. The SAS-TDMA algorithm dynamically builds a TDMA schedule for periodic messages and assigns radio channels to transmission nodes in response to events, i.e. requests for particular messages to be relayed from a source node to a sink node or a channel drop. Retransmissions of failed messages can be dropped when a pre-specified upper temporal bound is reached. The author's show that their algorithm minimizes the average TDMA schedule length and average latency of all scheduled packets and provides adaptation to the environmental conditions. Simulation results using the TOSSIM environment indicate that a network timely throughput of around 90% is achievable for a 100 node network and a choice of temporal upper bound of 8 seconds, although background overheads for aspects such as time synchronisation were not simulated). This provided clear improvements over a level-based approach and a node-based approach. Although more flexibility than that proposed in [4] is achievable, a drawback of the SAS-TDMA approach is the restriction that the period of any message must be restricted to be 7 superframes or less. This fixes the overheads of the scheduling algorithms at a manageable level, but it does not provide for a general solution that extends beyond sensor networks with near-uniform sample rates to generic factory automation and control applications. If the restriction is lifted, then scheduling overheads would become unmanageable and impractical due to the strong NP-hardness of generating TDMA schedules.

Considering more complex scheduling approaches than those based on simple cyclic polling, Liu et al. consider a generic framework for opportunistic scheduling in wireless networks [23]. They consider a time-slotted TDMA/CDMA (Collision Detection Multiple Access) access mechanism where both uplink and/or downlink transmissions are

scheduled in a superframe by a base (master) station. The uplink/downlink channels to each station (node) are assumed to be subject to time-varying interference, leading to variable Signal-to-Noise Ratio (SNR) and required transmission power. Scheduling policies are developed to maximize several specific objective functions: with respect to the current paper, temporal fairness is the relevant criteria to consider. It is shown that under the assumption that knowledge of the time-varying characteristics of the uplink/downlink channels is known, an optimal scheduling policy for their formulation can be obtained through solving a constrained optimization problem. Since exact knowledge of the time-varying characteristics of channels is not a practical assumption, an algorithm that can be implemented is also developed, making use of statistical estimation of unknown channel parameters.

Arguably, the closest work to the idea presented in this paper is the 'Feasible EDF' scheduler first presented by Shakkotai and Srikant in [19] and discussed in the introduction. In this scheduler, assuming accurate knowledge of the channel error state is known by the scheduler *prior* to scheduling a transaction, the *feasible* transaction with earliest deadline should be scheduled; such a policy has many desirable properties, and is optimal for an algorithm that uses one-slot channel state information [19][24]. However as reported in [19], there are a small number of circumstances in which a Feasible EDF may be outperformed by a scheduler using more than one-step channel state information, i.e. a scheduler which looks forward multiple steps into the future. Building upon this observation the feasible EDF method was extended by Kong & The [24] who presented a 'proactive' extension of the scheduling algorithm. The proactive extension requires knowledge of not only the current channel state but also future channel states and uses this information to help schedule packets that may otherwise miss their deadlines. It achieves this effect by dynamically expediting the deadlines of pending packets in anticipation of an upcoming degradation in channel quality, such that their deadline is moved to the latest point in time at which a successful transmission could take place (the original deadline is still used to remove expired packets from the run queue, however). A drawback of proactive EDF is that the process of advancing deadlines leads to priority inversions and blocking in the generated schedule, which may lead to domino-style deadline missed [25]. No formal analysis of the proposed proactive extension is given in [24], however simulation results indicate that it can in some circumstances have a small positive impact on timely throughput when compared to Feasible EDF.

A greater restriction is that since accurate knowledge of the current and future channel states is required in both the Feasible EDF and Proactive EDF schemes, the schedulers must be clairvoyant and hence they cannot be exactly implemented in practice [19][24]. This has led to research concentrating upon the on-line estimation of statistical channel parameters for prediction of channel states or the use of probe packets to assess channel quality prior to making scheduling decisions [19][24]. Whilst both modifications have been shown to produce good results, they can only ever approximate Feasible EDF or Proactive EDF in practice due to the inherent uncertainty in channel feasibility measurement, estimation and/or prediction [19][24]. Channel quality assessment techniques may also be complicated to implement on the network nodes and induce extra computational overhead and/or network bandwidth. In many industrial wireless applications the computational overhead on nodes, along with network bandwidth overheads and administration burdens, should be kept as low as is reasonably possible [14][22]. In addition, the Proactive EDF method in particular is highly sensitive to prediction errors – even when probe packets are employed - and performance is quick to deteriorate as the variance in estimating future channel states increases; this limits the applicability of the method to cases in which a channel has relatively short times between successive fades [24]. The technique to be

presented in this paper is based upon Feasible EDF but does not require channel state estimation and has a simple implementation with low-overheads. In addition, the scheduling overhead resides on the master node. Prior to the description of its operation, Section 3 describes the utilized system model.

3.0 System Models

3.1 Communication System Model

It is assumed that the communication system consists of a number of distributed nodes which share a common wireless medium to exchange messages in real-time. One node is designated the master node and controls the communications by sending requests to one of *M* slave nodes. Only a single slave node can normally be addressed in a master request[1]. The addressed slave node, upon receipt of the request from the master, processes the message and forms a response that is immediately transmitted back to the master. Slave nodes do not self-initiate transmission of messages. A transaction is formed from a consecutive pair of messages consisting of a master request (downlink) followed by a slave response (uplink). Transaction timings are assumed to be as shown in Figure 2. For each of the *N* slaves in the communication system, the downlink packet transmission time is assumed to have a worst-case duration of t_{rq} seconds and the uplink packet transmission time assumed to have a worst-case duration of t_{rs} seconds. The turn-around time in a slave is assumed to have a worst-case duration of t_{sp} seconds for each of the *N* slaves in the system. The turn-around time in the master is assumed to have a worst-case duration of t_{mp} seconds. Both of these latter timing measures are assumed to include transceiver turn-around time (if applicable) and packet processing overheads.

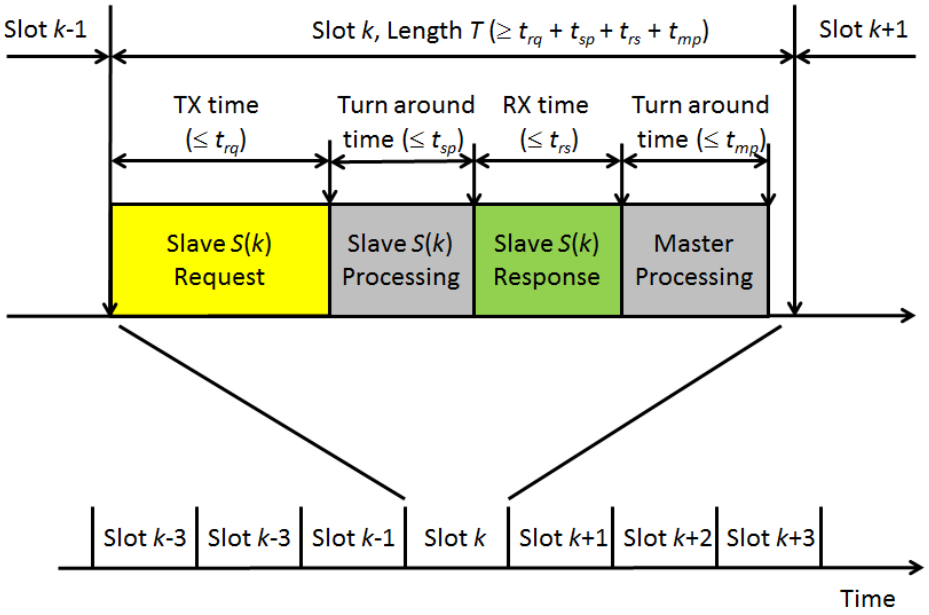

Fig. 2: Slotted schedule approach with sub-slot transaction timings

[1] Excepting that in protocols such as Modbus, some messages may be sent in broadcast mode in which no slave replies. This is compatible with the assumptions that follow, assuming that the Master node does not expect a reply in this specific case.

In the analysis that follows, it is taken that the system time is discrete and is divided into consecutive slots each having a uniform length of T seconds, such that $T \leq t_{rq} + t_{sp} + t_{rs} + t_{mp}$. System time is relative to an accurate clock located on the master node, and is indexed by a non-negative integer variable t. At run-time, the master dynamically assigns a transaction to the current slot t, and addresses the transaction request to the slave which is associated with that transaction (multiple disparate transactions may be associated with a single slave). Transactions may invoke either read or write operations in the slave nodes (or combinations thereof); the only restrictions are that the up/down link transmission and turnaround times specified above are respected by each transaction. The failure of either the request packet or response packet (or both) in any transaction is detectable by the master as an omission of the response at the end of the transaction slot. The master node is assumed to have full control over the number and timing of retransmission attempts through its ability to allocate transactions to slots mechanism. Note that only relatively minor modifications are needed to the above model to adapt to situations representing a more generic uplink/downlink scheduling framework, in which messages are dynamically scheduled into 'superframe' timeline slots by a centralised access point following transmission of a beacon. In this case, the slot length can be set to cover the worst-case length of either an uplink or a downlink packet, plus a short acknowledgement and turnaround time.

In the general case, the timing requirements for a set of real-time messages in an industrial master-slave (broadcast) communication system may be modelled by a set of N messages $\tau_1, \tau_2, \ldots, \tau_N$. Each message is represented by a 4-tuple:

$$\tau_i = (T_i, C_i, D_i, S_i) \tag{1}$$

In which T_i is the message period/inter-arrival, C_i is the worst-case transmission time of any instance of the message, D_i is the message relative deadline (see, e.g. [25]) and S_i is the slave ID. Given the discussion above, each transaction (request-response pair) may be represented in a similar fashion. In order to further simplify the model and analysis, it will be assumed that all transaction periods are integer multiples of the base slot time T, and that all relative deadlines are implicit (i.e. $T_i = D_i$). Since T is also the effective worst-case duration for a transaction, each transaction duration takes exactly one slot or less (i.e. $C_i = 1$), leading to a so-called 'Quantum' scheduling instance [26]. With this in mind we may model each transaction simply by its period T_i, and the slave ID S_i. For simplicity, let each slave ID satisfy $S_i \in \{1, 2, \ldots M\}$. The utilization of an individual transaction is given by $u_i = 1 / T_i$ and represents the fraction of time the network will be occupied processing the transaction over its lifetime (in the error-free case). Let the (error-free) utilization of the entire communication system be given as $U = \Sigma U_i$. Furthermore, since a slave may possibly be the subject of multiple disparate transactions, we may define the per-slave utilization U_i as follows:

$$\forall i, 1 \leq i \leq M : U_i = \sum_{S_i = i} \frac{1}{T_i} \tag{2}$$

The per-slave utilization represents the fraction of the time network will be occupied processing periodic transactions for a particular slave over its lifetime (in the error-free case). Although the traffic is assumed to be principally periodic as

detailed above, it is recognized that in many cases infrequent aperiodic transactions may also be required (e.g. for slave configuration updates in a SCADA system, such as an update the PID parameters in a slave controller). It is assumed that the master node may utilize any idle slots for these purposes. Again, only relatively minor modifications are needed to the above model to adapt to situations representing a more generic uplink/downlink scheduling framework. In this case, suppose that there are X slots of length T in each superframe, and let periods now be restricted to be integer multiples of XT (i.e. the superframe length). In such circumstances, each of the upcoming X superframe slots may be assigned to the slaves by the master only once every XT slots; since there will be no new transaction arrivals during the transmission of each superframe, this is not problematic [19]. If required the transmission of the beacon and control information may be modeled as a dummy transaction with period XT, since the time taken for transmission is usually less than the slot size.

3.2 Remarks

The system model presented above makes use of several simplifying assumptions. However, for most industrial packet radio protocols, there are limits on packet payload sizes (e.g. 128 bytes or even less) which are relatively short in comparison to wired protocols such as Ethernet; in addition, achievable data throughput may be high enough to give relatively short slot durations for many process control and factory automation applications [4][17-19]. A typical TIA-485 wireless modem may offer a data rate of 1 Mbps for short distances (300 m or less), and more typically 250 kbps for medium distances (3 km or less) [6]. With a maximum packet size of 32 bytes, and assuming a relatively fast slave turnaround time, then a slot lengths as low as T = 1 ms or T = 5 ms could easily be achieved. In previous works, a slot size of 20 ms was assumed for a 250 kbps wireless network operating at 2.3 GHz [4]; similar configurations have been used in [19]. Slot sizes of typical duration 5 ms - 15 ms are employed for industrial wireless protocols [11]). For many industrial automation and monitoring applications, ensuring periods are enforced to be the nearest whole multiple of such a slot size will not be an overly large restriction. To ensure that available payload is not unnecessarily wasted, transactions common to a slave may be merged in many cases, with the smallest period inherited. An efficient algorithm to pack signals into transactions while respecting maximum payload size can be found in [27]. Finally, it can be noted that if the model proposed in this paper is unduly restrictive for a given application, then other more complex general models for real-time scheduling a multiple-hop wireless link can be found in the literature [28].

3.3 Channel Error Model

As with previous work, the channel conditions for each master-slave link are assumed to be statistically independent, but not necessarily identically distributed. As previously mentioned, research has shown that packet level errors in wired and wireless communication links are likely to occur in transient bursts. A common way to model bursty behaviour in mobile and/or fading wireless links is to use a simple two-state discrete Markov model [29-31], such as is shown in Figure 3:

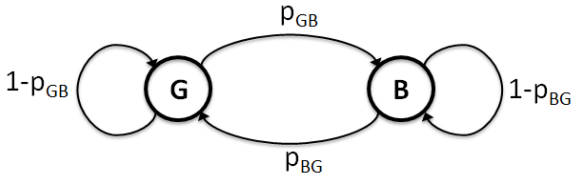

Fig. 3: Markov model for representing bursts of interference

The model has two states *G* and *B*, representing 'Good' and 'Bad' states respectively. With reference to the previous Section and the assumption of a slotted schedule, assume that the link between the Master and any particular slave at slot *k* can be represented by such a model, with the interpretation that if the link is in the Good state, a transaction attempt will be successful, and if the link is in the Bad state it will not be successful (lost). Transitions between the two states *G* and *B* have associated with them static probabilities p_{GB} and p_{BG}. The probability of remaining in a given state is then given by p_{GG} = 1-p_{GB} and p_{BB} = 1-p_{BG}. The model parameters p_{GB} and p_{BG} can be interpreted as follows: the reciprocal of p_{GB} defines the expected (mean) gap between error bursts μ_{EG}, and the reciprocal of p_{BG} defines the expected (mean) duration of error bursts μ_{EB}, both variables having a geometric distribution. The expected inter-arrival time of error bursts is then given as (μ_{EG} + μ_{EB}). Typically, it would be the case that μ_{EB} << μ_{EG}. As has been previously discussed by other authors, this model – despite its simplicity – can be used to great effect to model a Rayleigh-fading wireless channel [19][24].

Let the state of the link between the master and slave *j* at slot *k* be denoted by $l_j(k) \in \{$'*G*', '*B*'$\}$, and let the state of the Markov model at step *k* be encoded as the probability $s_j(k)$ that the link is in the error state, i.e. $s_j(k) = P\{l_j(k) =$ '*B*'$\}$. Applying the normal rules for Markov model state transitions, and assuming statistical independence between the slave link errors, then $s_j(k)$ depends only upon the previous state $s_j(k\text{-}1)$ and the transition probabilities p_{BB} and p_{GB}, and can be recursively computed according to:

$$s_j(k) = p_{BB} \cdot s_j(k-1) + p_{GB} \cdot (1 - s_j(k-1)) \tag{3}$$

Assuming that the state of the link at slot *t* - $s_j(t)$ - is known, the transient and steady-state evolution of the link state could be calculated using equation (3). The steady-state solution to the Markov chain is obtained by first setting $s_j(t) = s_j(k\text{-}1) = \pi_j$. Solving for π_j, we obtain that $\pi_j = p_{BG}/(1\text{-}p_{BB}\text{-}p_{GB})$, which gives the resting Packet Error Rate (PER) for link *j* (the probability that when the link is observed at some random sample time *t* we find that $l_j(t) =$ '*B*'). Following the acquisition of explicit knowledge about the link state at step *t* (such as by making an observation), the state transiently moves back to the steady state π with correlation (coherence) coefficient $\alpha_j = (p_{BB}\text{-}p_{GB})$ according to the relationship:

$$s_j(t+k) = \pi + (s_j(t) - \pi) \cdot \alpha_j^k \tag{4}$$

The special case in which errors do not arrive in bursts but only have a constant bit error rate of β is covered by setting $p_{BB} = p_{GB} = \beta$, after which we may see that $\alpha_j = 0$. Suppose that a packet error occurs during a transaction during slot *t*. Then, the link has been observed in the Bad state, and assuming that $\alpha > 0$ the probability of failure for an immediate retransmission is transiently higher than the resting PER π_j; equation (4) informs that a transient period (in terms of failure probability) will be entered and it will take some time for the link to recover to its resting level π. To ensure the

transient perturbation on the link probability decays by some factor $\delta > 0$, e.g. 0.001 (decay to 0.1% of initial value), since the transient decays as a geometric progression with coefficient α, in the ideal case any re-transmission should be delayed for a time gap of *g* slots according to [31]:

$$g = \lceil \log_{\alpha}(\delta) \rceil = \left\lceil \frac{\ln(\delta)}{\ln(\alpha)} \right\rceil \quad (5)$$

This would restore the link PER very close to its nominal level; however, since the deadline of the transaction would be much nearer (or even possibly elapsed) there is clearly a trade-off involved. However since the state of the link, the PER and the coherence coefficient α can never be accurately known (and the latter may also vary over time) – and in this paper it is assumed that no attempt is made to estimate them - then a 'good' (but not necessarily optimal) strategy to employ would be to delay subsequent re-transmission attempts by some small time gap to allow the link better chance to recover. This observation will be exploited in the design of the proposed Eligible EDF scheduling framework in the following Section.

4.0 Proposed Eligible EDF Scheduling Framework

In this Section, the proposed framework for scheduling periodic and aperiodic transaction transmissions and retransmissions will be described. Henceforth, the first transmission attempt of a transaction will be referred to as the *primary* transaction attempt, and should the primary transaction fail due to an error, subsequent retransmission attempts will be referred to as *backup* transaction attempts. The overall goal of Eligible EDF is to ensure that (i): every instance of every transaction is allocated a slot to attempt the primary transmission before its deadline elapses, (ii): as many backup transactions are processed before their deadline as possible, without violating the timing properties of any pending primary transactions and (iii) any aperiodic transactions are handled on a best-effort basis without interfering with primary or backup transmission attempts of pending periodic transactions.

4.1 General Scheduling Framework

From discussions in the previous Section, we may see that each transaction will arrive exactly once every T_i time slots. Let us further assume (without loss of generality) that all transactions are synchronous, i.e. their initial start times are all phased at $t = 0$. When transaction *i* arrives (becomes ready) at time slot *t*, its absolute deadline d_i is set at time $t + T_i$ and the transaction is moved into the primary ready queue. According to requirement (i), the scheduling procedure employed by the master node must allocate a slot to process the primary transaction in the interval $[t, d_i)$. For this, we use the EDF algorithm [19][25], in which the ready job with the nearest absolute deadline is always allocated the current slot. In the case of Quantum EDF scheduling, this algorithm is optimal and guarantees that all deadlines will be respected (in the error-free case) if and only if the condition $U \leq 1$ holds [26]. Suppose that the primary transaction is processed during some specific slot. If the transaction is successful, it is removed from the ready queue and scheduled for its next periodic release. If the transaction fails due to an error, then this is indicative (from the assumed error

channel model) that the link has entered the bad state. Then, considering requirement ii), it would be beneficial to consider scheduling a backup transaction to try to recover the situation before the deadline elapses. In addition, it is easy to see that it will not be beneficial to consider attempting any other transactions (to this particular slave) which have a *later* deadline than the considered (failed) transaction until either communication is re-established or the deadline of the considered (failed) transaction expires.

Therefore, we consider a scheduling framework that first orders pending transactions on a per-slave basis to ensure that Head of Line (HOL) transaction for each slave is the one with the closest temporal deadline. Suppose now that we associate with each slave a Boolean flag which indicates if the slave channel is *eligible* or *ineligible* for selection for transmission in the current slot. From amongst those slave channels which are deemed *eligible*, we select the slave with the closest temporal deadline for transmission. If the transaction is successful, it is removed from the slave ready queue and scheduled for its next periodic release. If the transaction is unsuccessful, however, the transaction is left in the queue and the slave channel is temporarily flagged as ineligible (and hence may be temporarily neglected from future scheduling decisions). According to requirement (ii), it is desired – ideally at some point before the deadline of the transaction elapses – to reset the slave channel as eligible once more such that a backup transaction may be attempted; this resetting must be done, however, in such a way that the overall communication network is not overloaded and other transactions are not forced to miss their deadlines due to 'domino-style' cascading timing violations. A simple but effective strategy to ensure temporal predictability will now be described.

4.2 Eligibility Replenishment

It is desired to replenish the ineligible slave channels - and hence re-queue backup transactions for possible transmission – at the earliest possible time, but in such a way that the timing requirements for any pending (but unprocessed) primary transactions on eligible slave channels are not jeopardized. For this purpose, an aperiodic server will be used to decide when a channel may be replenished [25]. The principal role of an aperiodic server is to allocate unused capacity in a periodic schedule to process aperiodic requests such that their waiting time is minimized. In the current context, the spare capacity of the overall wireless network is the utilization which is unused by the primary transactions in the error-free case, having the value (1-U). Although it is difficult to implement an optimal server in the general case [25], when Quantum scheduling is employed as is the current case the optimal server has a very simple form [26]. First, define the replenishment period of the server as the integer T_s, as follows:

$$T_s = \left\lceil \frac{1}{(1-U)} \right\rceil \tag{6}$$

If the time at which the last aperiodic transaction was attempted occurred at time t_l, the next pending aperiodic transaction may be *immediately* executed at any time slot equal to or subsequent to the time $t_n = t_l + T_s$, without impacting upon the timing properties of any primary transactions [26]. This server, with period selected according to (6), is related to the improved total bandwidth server for general EDF task scheduling (e.g. see [32]) and is optimal in the sense that the response times of any aperiodic transactions are minimized within the periodic schedule [26][32]. From

amongst those slave channels which are deemed *ineligible* at a particular time slot, if we select the slave with the transaction having the closest temporal deadline then this is the most urgent slave channel which should be considered for eligibility replenishment. Deadline ties should be broken by selecting the transaction with the smallest number of transmission attempts since its current release. If the eligibility of this slave is replenished at the earliest possible time by the server described above, then the most urgent backup transaction will be allocated the best possible service without overloading the remaining eligible slave transactions.

In addition to refreshing the eligibility of ineligible slaves, the server may just as well choose to insert an aperiodic request into the schedule if the server is eligible. Hence, to meet requirement (iii) if any aperiodic transactions arrive they can be held in a First In First Out (FIFO) queue and inserted into the schedule whenever a) the server is eligible and b) no slave channels currently require replenishment. This ensures that aperiodic requests are handled on a best-effort basis interfering with primary or backup transmission attempts of pending periodic transactions. Should any aperiodic transaction fail, then it may be re-inserted in the FIFO. The overall concept of the proposed scheduling framework is illustrated as shown in Figure 4.

From the Figure, it can be observed that a local clock in the master node is used to drive the main scheduler logic and the transaction release control mechanism. When a transaction becomes ready, it is inserted into the relevant slave queue which maintains the EDF ordering of transactions. The HOL transaction for each slave is passed to the slave eligibility logic, which maintains the eligibility status indicators, and also selects the eligible and ineligible transactions with the earliest deadlines using these indicators. The earliest eligible transaction is passed, along with an indicator that eligibility replenishment is required for any slave (or not), is transferred to the scheduling logic module. The scheduling logic module implements the server and decides, at the commencement of each slot, whether the server is eligible – and if so, whether to implement a slave channel replenishment or insert an aperiodic transaction from the FIFO queue ahead of the earliest eligible HOL transaction to the slot transmission / reception handling logic. Based upon the success (or otherwise) of this transmission, the channel eligibility is updated or a failed aperiodic is re-queued. Implementation of the EDF queues and management of the periodic transactions can be done very efficiently (see, e.g. [33]), leading to a solution which has minimal overheads which resides mainly in the master node. Slave nodes only need implement the communication protocol itself, along with simple Operating System (OS) mechanisms (e.g. see [14]) in addition to the task application software.

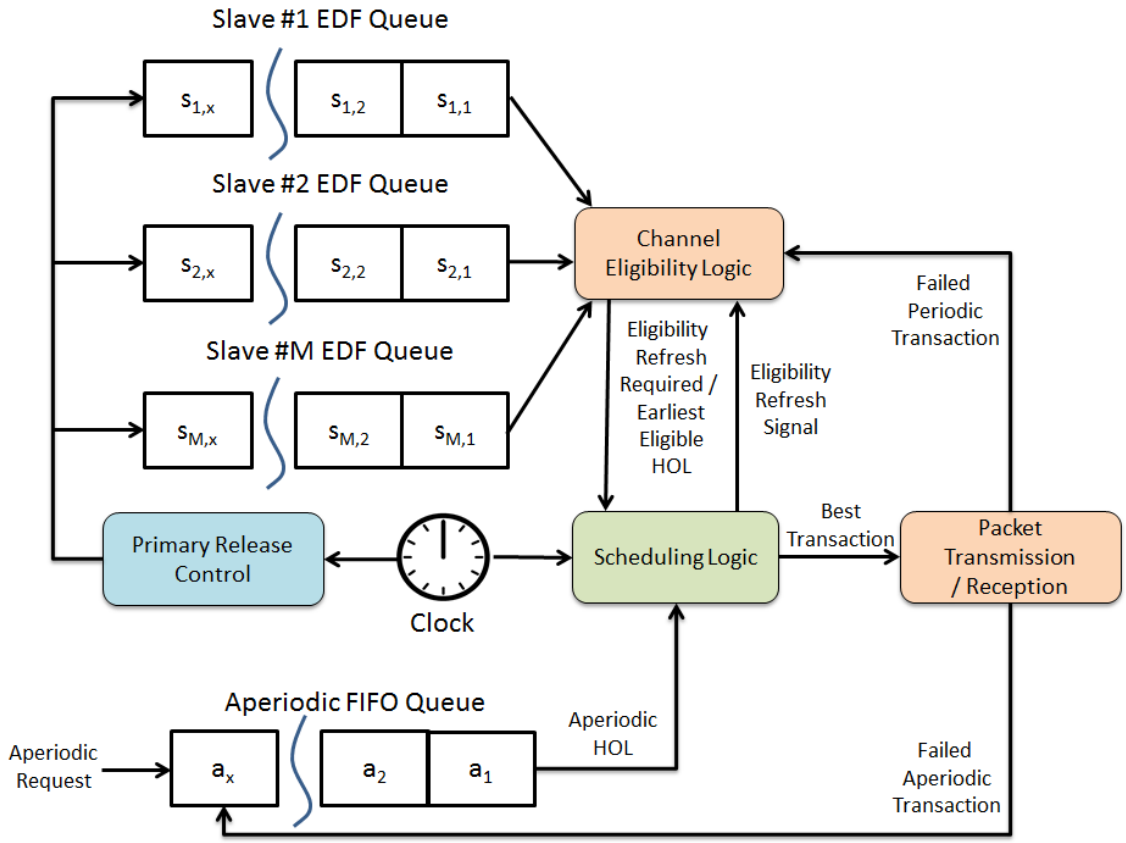

Fig. 4: Proposed Eligible EDF scheduling framework with server-based eligibility refreshment and aperiodic transaction handling

From the descriptions above, it may be observed that when a link transits to the error state, then subsequent attempts at a failed transaction will typically be separated by a duration of at least T_s timeslots each, and simultaneously progress will still be made on transactions to slaves which have not been flagged as entering an error state. This approach has several interesting properties; by inserting a gap of at least T_s slots between subsequent retransmission attempts, for the channel model described in Section 3.3 the negative transient reduction in the probability of success following each failure decays by a factor α^{Ts}. Thus, Eligible EDF gives an approximation to Feasible EDF with unused channel bandwidth dynamically allocated to the purpose of effectively recovering slave channels.

4.3 Configuration

If a rough estimation of the slave channel PERs are available, this can be used to give a useful lower bound on the amount of unused channel bandwidth that is required to ensure stable steady-state operation of the network under errors. Given knowledge of the per-slave utilization U_i from (2) and the slave PER π_i as discussed in Section 2.3, then the following condition is required to hold:

$$\sum_{i=1}^{M} U_i \cdot (1+\pi_i) \le 1 \tag{7}$$

This condition only gives a guarantee that, asymptotically, the network will have enough steady-state capacity to cover errors: no guarantees are given as to whether individual errors can be effectively recovered before or after their respective transaction deadlines. As will be demonstrated in Section 5, the mean burst length relative to the minimum relative deadline influences the number of deadline misses and hence the timely throughput of the network. To provide statistical guarantees of timeliness, then clearly some additional factor-of-safety to scale above the mean PER in each channel is required in equation (7). The analysis of such a safety factor and the resulting effect on network reliability is beyond the scope of the current paper but forms an interesting area of future work.

5.0 Simulation-Based Evaluation

In this Section, initial experiments to evaluate the performance of the proposed Eligible EDF scheme against three related schemes are considered. A small network with a limited range of transaction periods is chosen to simplify the simulation; a relatively aggressive environment was also chosen to ensure meaningful results might be obtained in a relatively short time. The section begins with a short description of the simulation environment.

5.1 Simulation Environment

In order to provide the comparative performance analysis of the proposed scheme to other related schemes, custom simulation software was developed for a desktop PC. Both wired and wireless networks can be evaluated using a variety of different methods, which may be broadly categorized into theoretical analysis, simulation, emulation, virtualization and direct measurement in a real world testbed [34]. Although each method has its own advantages and disadvantages, simulations provide a high level of experimental control, produce repeatable results, and are highly flexible and low cost [34]. For these reasons, the simulation-based approach was chosen. The principal disadvantage of simulation, however, is that abstractions of reality are invariably needed, leading to simplifications and possible deviations from real-world measurements [34]. The principal measurements of interest to be taken in this paper are comparative measures of the timely throughput of the network under different master node scheduling algorithms (refer to section 5.4 below). In addition, the proposed technique is technology-agnostic, save for the assumption of a slotted schedule within a master-slave arrangement. As such, there is no requirement to simulate specific network protocols, communication stacks or node hardware, and the simulation requires only simple (but representative) channel error models and deterministic scheduling algorithms. Application level simulation was chosen as an appropriate level of abstraction; application level simulators are high-level simulations that are generally unaware of specific OS and hardware. As such they feature short simulation times at the cost of potentially no code reuse between simulation and target system; the latter is not a concern of this paper.

The master node scheduling algorithms, channel error models and slave processing algorithms used in the experiments were coded in C++ and executed in a simulation environment running on a standard desktop PC. Elements of this environment have previously been described by the author and successfully employed to carry out detailed statistical evaluations of industrial communication networks [31]; in addition, a custom architecture of similar design (albeit with more advanced features) has recently been documented for similar purposes by others [34]. The overall structure of the simulation is as shown in Figure 5.

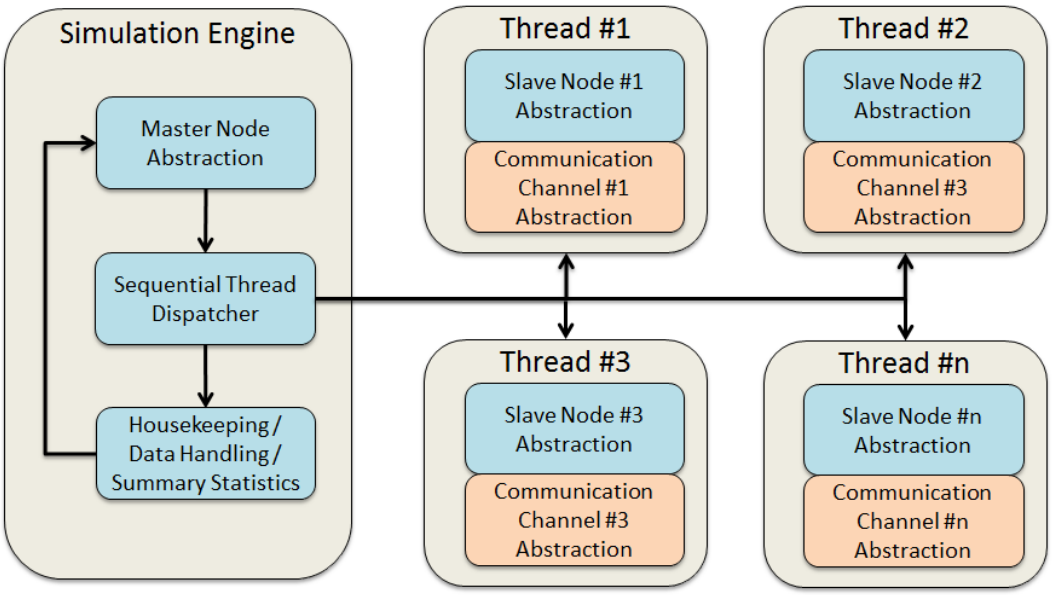

Fig. 5: Simulation Software Architecture

The simulation engine acts as the time base for the simulation, implements the main program loop, handles/stores data and calculates running summary statistics to display to the user. The main program loop directly simulates the master node abstraction and the scheduling algorithm under simulation, and sequentially dispatches threads that implement the slave node abstractions and communication channel models. Shared variables are used for the transfer of data sand event signaling between the main loop and the threads.

5.2 Network Configuration

A master-slave system with one master and N = 5 slaves in a small work cell is assumed. A total of 10 Transactions form the communication requirements, as shown in Table I below. Periods are shown in terms of slot sizes, which were taken to be 10 ms in duration. Also indicated are the total utilization U and the server replenishment period T_s as calculated using (2). In effect, a quarter of the network utilization can be safely allocated for handling secondary transactions, i.e. one slot out of every 4 during periods of interference with periodic tasks also pending.

5.3 Interference Configuration

To generate link interference, the Markov model of Figure 3 was employed to simulate interference and fading on each channel between the Master and Slaves. This was done to simulate intermittent interference affecting the slaves in an unpredictable manner, which was also not uniform across all slaves simultaneously. Ten different link configurations were considered, as follows. In the former five configurations the links had an overall slotted packet error rate (PER) of 0.1, but with burst distribution having mean lengths of 2, 4, 6, 8 and 10 slots respectively. Although the PER was identical in each configuration this resulted in a different coherence co-efficient for each of the five different burst lengths, with the co-efficient increasing with increasing mean burst length. In the latter five configurations, the links had an overall slotted packet error rate (PER) of 0.01. Similar distributions of burst length were also considered in these five cases. Although not all nodes are affected by bursts simultaneously, it is clear that both these levels of PER represent

aggressive environments; especially in the case of longer burst lengths, it is clear that deadlines will inevitably be missed. In all experimental configurations, it is easy to see that relationship (7) holds.

Table I: Communication requirements

Transaction ID	Slave ID	Period T_i	Utilization U_i
1	1	10	0.1000
2	2	10	0.1000
3	3	10	0.1000
4	4	10	0.1000
5	5	15	0.0667
6	5	15	0.0667
7	1	20	0.0500
8	2	20	0.0500
9	3	40	0.0250
10	3	100	0.0100
		U:	0.6683
		T_s:	4

5.4 Experiment Configurations

The simulations were configured as described above and in each configuration the network was simulated for time duration of 10^6 slots. During the simulations, the number of primary transactions generated was recorded along with the number of transactions (primary or backup) affected by errors and the number of deadline misses. For comparative purposes, three additional scheduling methods were simulated along with the proposed scheduling technique. The four methods were as follows:

- The 'Eligible' EDF scheduler, with server-based replenishment method, as described in Section 4.

- A 'Persistent' EDF scheduler, that immediately re-queued any failed primary transactions in the ready queue with its original deadline. The persistent re-queuing was allowed to continue indefinitely in the event of backup failure(s) until the deadline of the failed transaction had elapsed.

- A 'Feasible' EDF scheduler as described in [19], which as discussed in the introduction assumes full knowledge of the upcoming channel states before making scheduling decisions. The QEDF/F results are given for benchmark purposes, as the algorithm cannot be exactly implemented in practice. An accuracy of 100% for channel state estimation was assumed for each slave.

- A 'Lazy' EDF scheduler, which did not re-queue any failed primary transactions following an error. Hence, no attempts at backup transaction transmissions were made after primary failure.

In all cases, 6,683,334 primary transactions were generated during each simulation. The same initial condition was employed by the pseudorandom number generator in each of the simulations across the different burst configurations.

5.5 Results

The results obtained for each scheduler in the configurations having PER of 0.1 and 0.01 are shown in Tables II and III respectively. The Tables display the number of deadline 'hits', the number of deadline 'misses' and the number of retransmission attempts (retries) made by each scheduler in each case. Note that the number of retries is zero for both the Feasible EDF and Lazy EDF schedulers; the former since no attempt is made to transmit over channels in error (hence no retries are needed), and the latter by deliberate design (none are attempted since the scheduler gives up). Also indicated in each table is the overall effectiveness of each approach, represented as the probability of deadline hit as a Binomial proportion. For completeness, upper and lower 95% confidence intervals for this Binomial proportion are also shown in these Tables. The intervals were calculated exactly using the method described in [35]. Due to the large number of trials involved, the intervals were found to be extremely tight; for each configuration, the intervals did not overlap across each scheduler. This allows judgments regarding the relative effectiveness of each scheduling approach to be made with high confidence.

Table II: Results for PER = 0.1

Mean Burst Length	Scheduler	Hits	Misses	Retries	p^-	p	p^+
2	Feasible EDF	6680376	2958	0	0.99954	0.99956	0.99957
	Eligible EDF	6669066	14268	888777	0.99783	0.99787	0.99790
	Persistent EDF	6652684	30650	1212569	0.99536	0.99541	0.99547
	Lazy EDF	6015212	668122	0	0.89980	0.90003	0.90026
4	Feasible EDF	6625700	57634	0	0.99131	0.99138	0.99145
	Eligible EDF	6599311	84023	1150807	0.98734	0.98743	0.98751
	Persistent EDF	6405516	277818	1876690	0.95828	0.95843	0.95858
	Lazy EDF	6015400	667934	0	0.89983	0.90006	0.90029
6	Feasible EDF	6550754	132580	0	0.98006	0.98016	0.98027
	Eligible EDF	6521750	161584	1318230	0.97571	0.97582	0.97594
	Persistent EDF	6171055	512279	2211672	0.92315	0.92335	0.92355
	Lazy EDF	6015800	667534	0	0.89989	0.90012	0.90035
8	Feasible EDF	6484860	198474	0	0.97017	0.97030	0.97043
	Eligible EDF	6457293	226041	1429936	0.96604	0.96618	0.96632
	Persistent EDF	5995562	687772	2406974	0.89686	0.89709	0.89732
	Lazy EDF	6015301	668033	0	0.89982	0.90004	0.90027
10	Feasible EDF	6431881	251453	0	0.96223	0.96238	0.96252
	Eligible EDF	6406347	276987	1508097	0.95840	0.95856	0.95871
	Persistent EDF	5863084	820250	2529264	0.87702	0.87727	0.87752
	Lazy EDF	6015581	667753	0	0.89986	0.90009	0.90031

Table III: Results for PER = 0.01

Mean Burst Length	Scheduler	Hits	Misses	Retries	p^-	p	p^+
2	Feasible EDF	6682997	337	0	0.99994	0.99995	0.99995
	Eligible EDF	6682750	584	88571	0.99991	0.99991	0.99992
	Persistent EDF	6682203	1131	121074	0.99982	0.99983	0.99984
	Lazy EDF	6616837	66497	0	0.98997	0.99005	0.99013
4	Feasible EDF	6677560	5774	0	0.99911	0.99914	0.99916
	Eligible EDF	6676534	6800	122255	0.99896	0.99898	0.99901
	Persistent EDF	6665570	17764	195427	0.99730	0.99734	0.99738
	Lazy EDF	6616856	66478	0	0.98998	0.99005	0.99013
6	Feasible EDF	6670072	13262	0	0.99798	0.99802	0.99805
	Eligible EDF	6668827	14507	144733	0.99779	0.99783	0.99786
	Persistent EDF	6644239	39095	239094	0.99409	0.99415	0.99421
	Lazy EDF	6616758	66576	0	0.98996	0.99004	0.99011
8	Feasible EDF	6663713	19621	0	0.99702	0.99706	0.99711
	Eligible EDF	6662464	20870	157154	0.99683	0.99688	0.99692
	Persistent EDF	6626466	56868	261933	0.99142	0.99149	0.99156
	Lazy EDF	6617202	66132	0	0.99003	0.99010	0.99018
10	Feasible EDF	6657811	25523	0	0.99613	0.99618	0.99623
	Eligible EDF	6656616	26718	169685	0.99595	0.99600	0.99605
	Persistent EDF	6609732	73602	283156	0.98891	0.98899	0.98907
	Lazy EDF	6616252	67082	0	0.98989	0.98996	0.99004

In order to allow easier visualization of the obtained data, the hit probability for each scheduler was plotted as a function of the mean burst length for a PER of 0.1 in Figure 6 and for a PER of 0.01 in Figure 7.

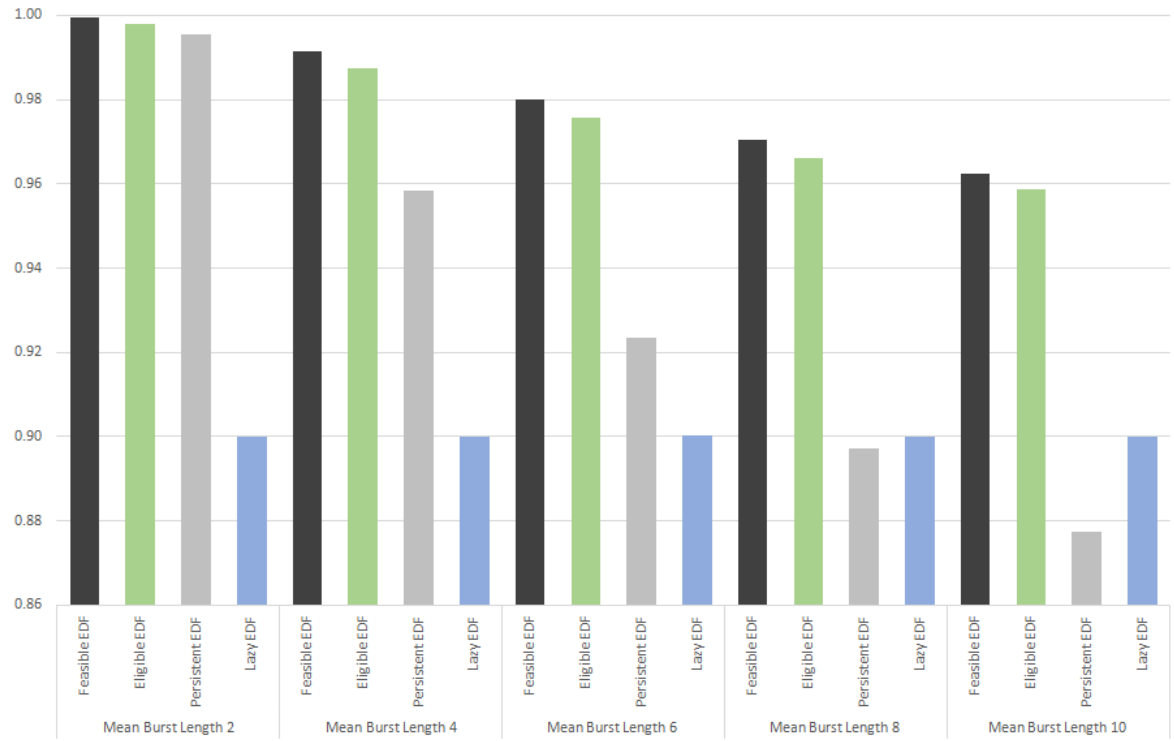

Fig. 6: Probability of successfully meeting deadlines for each scheduler for PER = 0.1

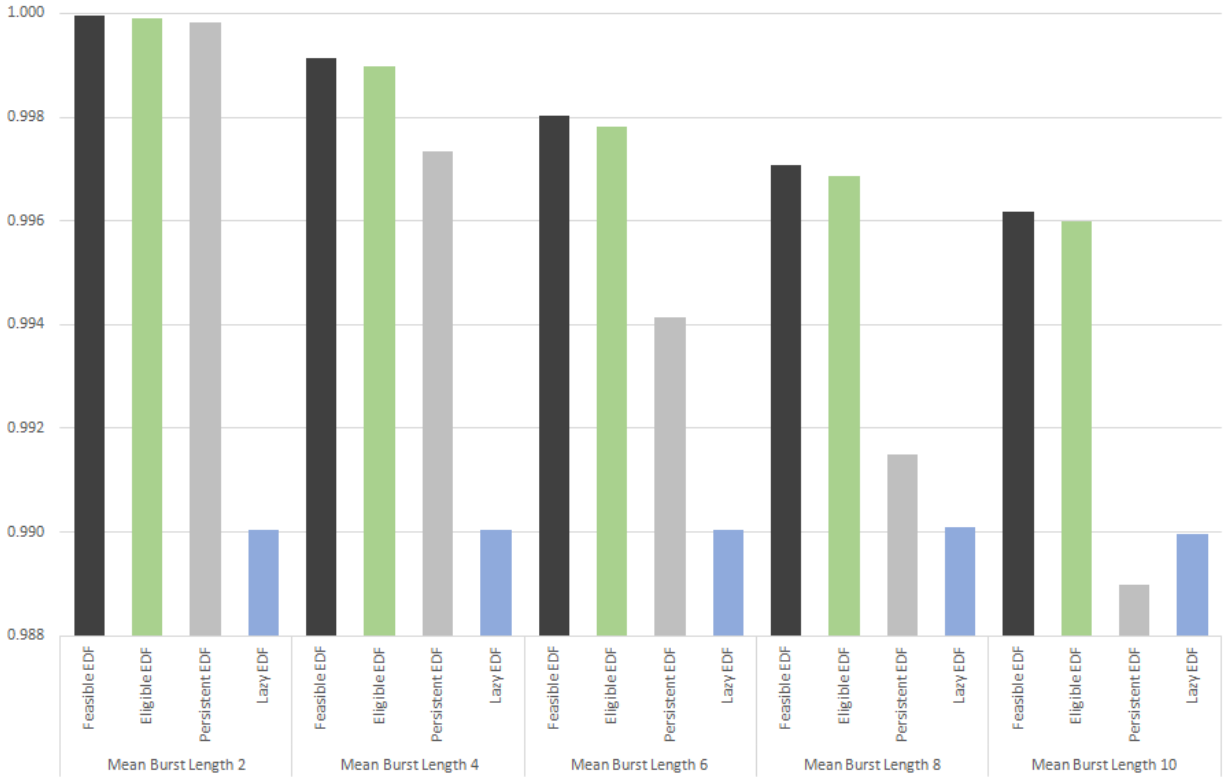

Fig. 7: Probability of successfully meeting deadlines for each scheduler for PER = 0.01

5.6 Analysis and Discussion

For all cases of mean burst duration, it may be observed from Figures 6 and 7 that the choice of scheduling technique had a clear influence upon the number of deadline misses and overall effectiveness of the communication system during the course of the simulations. At both PER levels, the mean burst duration had a significant influence upon the hit probability across each scheduler. For all cases of mean burst length, the hit probability of Feasible EDF is slightly better than Eligible EDF; both are consistently better than both Persistent and Lazy EDF. For the smallest level of mean burst length (2), the performance of Feasible, Eligible and Persistent EDF is very similar; as the burst length increases, the hit probability of Eligible EDF relative to Feasible EDF remains approximately consistent but both experience a decline. However, the hit probability of Persistent EDF declines much more rapidly relative to Feasible EDF and Eligible EDF as the burst length increases. For both levels of PER, the performance of Persistent EDF deteriorates below the hit probability of Lazy EDF when the mean burst length is equal to 10 (the value of the smallest relative deadline), indicating that the recovery mechanism is causing more problems than it is solving. Lazy EDF achieves consistent performance in terms of hit probability across all configurations and achieves a value approximately equal to (1-PER). Thus from inspection of these data points, it can be summarized that for these experiments (i) Lazy EDF is the least sensitive to burst length but consistently gives sub-optimal performance, (ii) Persistent EDF is the most sensitive to increasing burst length but gives close to optimal performance at shorter burst lengths and (iii) both Feasible EDF and Eligible EDF exhibit lower sensitivity to burst length than Persistent EDF, leading to graceful degradation.

Inspection of the Tables also reveals that the number of retries was also influenced by the PER, the mean burst duration and the choice of scheduler. From Tables II and III it may be observed that the Eligible EDF approach consistently attempted fewer retries than the Persistent EDF approach across all configurations, indicating that better use was made of the available slack time in the schedule. To obtain further insights into this and other aspects of the averaged performance, Table IV below displays a summary of each scheduling approach. Displayed are the average across all experiments of the relative optimality of each approach (in comparison to the Feasible EDF approach) and also the average across all experiments of the error recovery effectiveness. The former metric gives an indication of the overall timely throughput of the scheduler relative to the optimal timely throughput for the given channel conditions. The latter metric gives an indication of the overall effectiveness of the error recovery mechanism of each scheduling technique, and is reported as the percentage of the number of errors that the scheduler was able to recover once affected by errors. For Feasible EDF, this figure is not applicable since no transmission errors were present due to the clairvoyant mode of operation avoiding transmission over a channel in the error state. As discussed, this assumption that the Feasible EDF scheduler has explicit knowledge of the future channel error states made available before scheduling decisions are made is not practical for real implementations. Nevertheless, the baseline results for this scheduler – while not achievable in practice - allows the level of sub-optimality of the other approaches to be numerically quantified in each case.

Table IV: Summary Statistics for Each Approach

Scheduler	Relative Optimality (%)	Error Recovery Effectiveness (%)
QEDF/F	100.00	N/A
QEDF/E	99.81	78.15

QEDF/P	97.22	36.84
QEDF/L	95.49	0.00

In summary, on average the proposed Eligible EDF scheduler achieved a timely throughput of 99.81% of the optimal achievable timely throughput, which was a greater than 2% increase over the nearest rival which was persistent EDF achieving 97.22%. The effecvieness of the error recovery mechanism of Eligible EDF was 78.15%, which was over twice the value of the closest rival which again was persistent EDF at 36.84%. These results give an indication that over the range of PER levels and mean burst length characteristics that were considered, Eligible EDF made better use of the available bandwidth than Persistent EDF and Lazy EDF and that the selectivity in terms of the number of retransmissions (and their timing) introduced by the server-based mechanism in Eligible EDF increased the hit probability (timely throughput) and led to a more effective error recovery performance. Although the burst distribution had an impact upon the achievable optimal performance, the proposed Eligible EDF approach was more consistently able to achieve a hit probability close to that achievable with a clairvoyant Feasible EDF scheduler, without requiring channel state estimation. Recalling that it was assumed that the channel state could be estimated with 100% accuracy in the Feasuible EDF scheduler - an unrealistic assumption - the performance obtained for Eligible EDF seems adequate for indstrial applications. As a final comment, it is noted that for industrial applications requiring high network availabilty, the proposed Eligible EDF scheduler may easily be employed within a parallel redundancy framework (e.g. see [8]) by using ‘OR’ logic and voting applied to slave responses. An interesting area for future exploration will be to test the sensitivity of the channel estimation assumptions of the Feasible EDF approach, to gauge the level at which the performance gain over Eligible EDF is lost. This is beyond the scope of the current paper and is left for future work.

6.0 Summary and Future Work

With increasing penetration of wireless networks into industrial applications, reliable scheduling of transmissions and retransmissions in a master-slave wireless networks in order to meet real-time deadlines in the presence of interference is a pertinent subject. It was suggested in this work that in many practical cases, the use of a server-based Eligible EDF scheduling framework on the master node to assign transmissions and retransmissions to the available schedule slots gives close to the (optimal) performance which can be obtained by a previously described clairvoyant EDF scheduler. Data obtained from a series of representitive simulation-based experiments indicated that the level of timely throughput was, on average, 99.81 % of the optimal achievable timely throughput. In addittion to the areas of future work related to statistical timing guarantees and explorations of channel estimation sensitivity which have already been identified in the paper, more detailed experimental investigations using representative hardware are also planned to further explore the performance of the proposed method.

Acknowledgements

The author wishes to than the anonymous reviews for their constructive feedback on the initial and subsequent manuscript drafts, which helped to improve the final version of the paper considerably.

References

[1] A. Flammini, P. Ferrari, D. Marioli, E. Sissini & A. Taroni. Wired and wireless sensor networks for industrial applications. *Microelectronics Journal*, Vol. 40, pp. 1322-1336, 2009.

[2] H. A. Thompson. Wireless and Internet communications technologies for monitoring and control. *Control Engineering Practice*, Vol. 12, No. 6, pp. 781-791, 2004.

[3] W. Ikram & N.F. Thornhill. Wireless Communication in Process Automation: A Survey of Opportunities, Requirements, Concerns and Challenges. In: *Proceedings of the 8th IFAC Conference on Control (UKACC)*, Coventry, UK, September 2010.

[4] G. Gamba, F. Tramarin & A. Willig. Retransmission Strategies for Cyclic Polling Over Wireless Channels in the Presence of Interference. *IEEE Transactions on Industrial Informatics*, Vol. 6, No. 3, pp. 405-415, 2010.

[5] A. Abdrabou and A.M. Gaouda. Considerations for packet delivery reliability over polling-based wireless networks in smart grids. *Computers and Electrical Engineering*, Vol. 41, pp. 368–382, 2015.

[6] C. Goldman. Advantages of wireless I/O. *Rockwell Automation: The Journal*, Vol. 17, No. 5, pp. 38-42, October 2010.

[7] A. Frotzscher, U. Wetzker, M. Bauer, M. Rentschler, M. Beyer, S. Elspass and H. Klessig. Requirements and current solutions of wireless communications in industrial automation. In: *Proceedings of the IEEE International Conference on Communications (ICC)*, Sydney, Australia, June 2014.

[8] M. Hendawy, M. ElMansoury, K.N. Tawfik, M.M. ElShenawy, A.H. Nagui, A.T. ElSayed, M. Rentschler, H.H. Halawa, R.M. Daoud, H.H. Amer and H.M. ElSayed. Application of Parallel Redundancy in a Wi-Fi-based WNCS using OPNET. In: *Proceedings of the 27th IEEE Canadian Conference on Electrical and Computer Engineering (CCECE)*, September 2014.

[9] I.F. Akyildiz, W. Su, Y. Sankarasubramaniam and E. Cayirci. A survey on sensor networks. *IEEE communications magazine*, Vol. 40, No. 8, pp. 102-114, 2002.

[10] B. Rashid and M. H. Rehmani. Applications of wireless sensor networks for urban areas: a survey. *Journal of Network and Computer Applications*, Vol. 60, pp. 192-219, 2016.

[11] D. Christin, P.S. Mogre and M. Hollick. Survey on Wireless Sensor Network Technologies for Industrial Automation: The Security and Quality of Service Perspectives. *Future Internet*, Vol. 2, pp. 96-125, 2010.

[12] A. Ajith Kumar S., K. Øvsthus and L.M. Kristensen. An Industrial Perspective on Wireless Sensor Networks — A Survey of Requirements, Protocols, and Challenges. *IEEE Communications Surveys & Tutorials*, Vol. 16, No. 3, pp. 1391-1412, 2014.

[13] N.A. Pantazis, S.A. Nikolidakis and D.D. Vergados. Energy-efficient routing protocols in wireless sensor networks: A survey. *IEEE Communications Surveys & Tutorials*, Vol. 15, No. 2, pp. 551-591, 2013.

[14] C.-C. Chang, C.-B. Lin. A timer-based operating system for ZigBee sensor platforms. *Computers and Electrical Engineering*, Vol. 56, pp. 509-518, November 2016.

[15] F. Akhtar and M. H. Rehmani. Energy replenishment using renewable and traditional energy resources for sustainable wireless sensor networks: A review. *Renewable and Sustainable Energy Reviews*, Vol. 45, pp. 769-784, 2015.

[16] M. Short, F. Abugchem and U. Abrar. Dependable Control for Distributed Wireless Control Systems. *Electronics*, Vol. 4, No. 4, pp. 857-878, 2015.

[17] S. Gobriel, R. Cleric and D. Mosse. Adaptations of TDMA scheduling for Wireless Sensor Networks. In: *Proceedings of the 7th International Workshop on Real-Time Networks*, July 2009.

[18] R. Costa, P. Portugal, F. Vasques and R. Moraes. A TDMA-based Mechanism for Real-Time Communication in IEEE 802.11e Networks. In: *Proceedings of the 15th IEEE International Conference on Emerging Technologies and Factory Automation (ETFA)*, 2010.

[19] S. Shakkotai and R. Srikant. Scheduling Real-Time Traffic With Deadlines over a Wireless Channel, *Wireless Networks*, Vol. 8, pp. 13-26, 2002.

[20] E. Toscano and L. Lo Bello. Multichannel Superframe Scheduling for IEEE 802.15. 4 Industrial Wireless Sensor Networks. *IEEE Transactions on Industrial Informatics*, Vol. 8, No. 2, pp. 337-350, 2012

[21] The Modbus Organization. *Modbus Application Protocol Specification v1.1b3*. Technical Report, April 2012, available electronically at: http://www.modbus.org [Accessed October 2016].

[22] W. Shen, T. Zhang, M. Gidlund and F. Dobslaw. SAS-TDMA: a source aware scheduling algorithm for real-time communication in industrial wireless sensor networks, *Wireless Networks*, Vol. 19, No. 6, pp. 1155-1170, 2013.

[23] X. Liu, E.K.P. Chong, N.B. Shroff. A framework for opportunistic scheduling in wireless networks, *Computer Networks*, Vol. 41, pp. 451-474, 2003.

[24] P.-Y. Kong and K.-H. The. Performance of proactive earliest due date packet scheduling in wireless networks. *IEEE Transactions on Vehicular Technology*, Vol. 53, No. 4, pp. 1224 – 1234, 2004.

[25] G. C. Buttazzo. *Hard Real-Time Computing Systems—Predictable Scheduling Algorithms and Applications*. Kluwer Academic Publishers, 1997.

[26] P.G. Jansen, F. Hanssen and M.E. Lijding. Scheduling of Early Quantum tasks. In: *Proceedings of the 15th Euromicro Conference on Real-Time Systems (ECRTS)*, 2003.

[27] F. Polzbauer, I. Bate and E. Bremner. Optimized Frame Packing for Embedded Systems. *IEEE Embedded Systems Letters*, Vol. 4, No. 3, pp. 65-68, 2012.

[28] M. Jonsson and K. Kunert. Towards Reliable Wireless Industrial Communication with Real-Time Guarantees. *IEEE Transactions on Industrial Informatics*, Vol. 5, No. 4, pp. 429-442, 2009.

[29] E.N. Gilbert. Capacity of a burst-noise channel. *Bell Systems Technical Journal*, Vol. 39, pp. 1253–1265, 1960.

[30] M. Yajnik, S. Moon, J. Kurose and D. Towsley. Measurement and Modelling of the Temporal Dependence in Packet Loss. In: *Proceedings of INFOCOM,* Vol. 1, pp. 345-352, 1999.

[31] M. Short, I. Sheikh, I. & S.A.I. Rizvi, A Window Transmission Technique for CAN Networks. *Journal of Systems Architecture: Embedded Software Design*, Vol. 69, pp. 15-28, September 2016.

[32] M. Spuri and G.C. Buttazzo. Scheduling aperiodic tasks in dynamic priority systems. *Real-Time Systems*, Vol. 10, No. 2, pp. 179–210, 1996.

[33] M. Short. Improved task management techniques for enforcing EDF scheduling on recurring task sets. In: *Proceedings of the 16th IEEE Real-Time and Embedded Technology and Applications Symposium (RTAS)*, pp. 56-65, Stockholm, Sweden, April 2010.

[34] Ericsson, N., Lennvall, T., Akerberg, J. and Bjorkman, M. Custom simulation of Industrial Wireless Sensor and Actuator Networks for improved efficiency during Research and Development. In: *Proceedings of the 22nd IEEE International Conference of Factory Automation and Emerging Technologies (ETFA)*, Limassol, Cyprus, September 2017.

[35] J.T. Morisette and S. Khorram. Exact Binomial Confidence Interval for Proportions. *Photogrammetric Engineering & Remote Sensing*, Vol. 64, No. 4, pp. 281-283, April 1998.

Author Biography

Michael Short holds B.Eng. and Ph.D. degrees from the University of Sunderland. He worked at the University of Sunderland and the University of Leicester, and is now a Reader at Teesside University. His academic research interests include embedded systems, communications, control and smart grid applications. He is author or co-author of more than 100 reviewed technical papers and articles.